INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 8, Issue 2 - V8I2-1262)
Available online at: https://www.ijariit.com

Snowflake Internship/Training Project Report

Y. Swetha
swethayuvaraj00@gmail.com
Gandhi Institute of Technology and Management, Visakhapatnam, Andhra Pradesh

Dr. G Sandhya Devi
sgogula@gitam.edu
Gandhi Institute of Technology and Management, Visakhapatnam, Andhra Pradesh

ABSTRACT

Snowflake is a SaaS-based data warehouse (DWH) platform that runs over an AWS or MS Azure cloud infrastructure. (You might hear this called data warehouse as a service.) Unlike other warehouse solutions, Snowflake utilizes an enhanced ANSI-compliant SQL engine that is designed to work solely on the cloud.

Keywords— *Snowflake, Secure Data Share, Time Travel, Dynamic Data Masking, Loading Data into Table using Web User Interface, Roles and Privileges*

1. INTRODUCTION

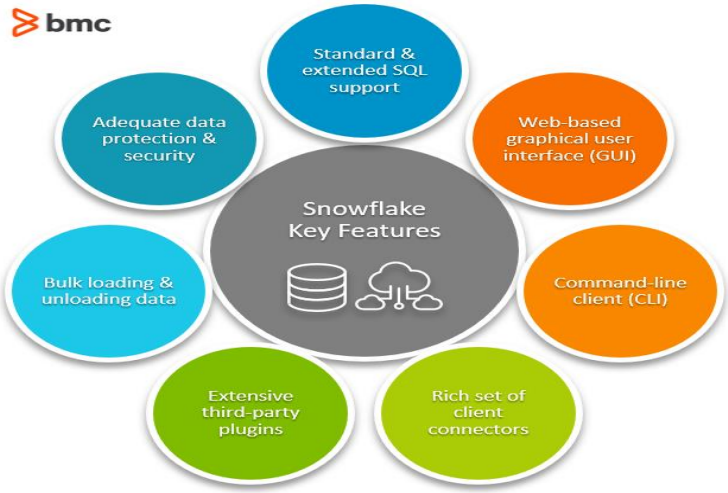
Fundamentally, Snowflake's core architecture enables it to run on the public cloud, using virtual compute instances and efficient storage buckets, making it a highly scalable and cost-efficient solution to process enormous amounts of big data.

The Snowflake technology Data Warehouse is particularly useful for companies looking for a platform that offers unique solutions that a traditional data platform cannot. In addition, their convenience and capability have made it all but unnecessary for enterprises to set up their own data warehouses. Snowflake provides a number of interfaces that you can use to write an application that connects to Snowflake.

These interfaces support different programming languages and development platforms. Snowflake also provides a number of ways in which you can extend SQL and the built-in, system-defined functions provided by Snowflake. You can write your own functions and procedures that you can call from SQL.

The objective of this snowflake training is

- To impart knowledge on SNOWFLAKE data warehouse cloud to the students of computer engineering colleges.
- This course is for a candidate who wants to learn snowflakes from scratch and work through a fully functional Project. This course is apt for those who have been working on snowflake and intend to move the expertise to the future of business intelligence.
- Key features of Snowflake: When compared to legacy DWH technologies, Snowflake offers a number of features, including:



2. ABOUT ORGANIZATION

Kipi an Apisero™ company on a mission to help customers deliver value and enable their businesses to thrive. From start-to-finish, we're committed to ensuring your business has the tools and support it needs to drive powerful data analytics and insights. Our founders launched kipi.bi to help businesses overcome data gaps and deliver rapid insights at scale. As an Apisero™ Company, kipi.bi carries Apisero's deep rooted commitment to delivering quality solutions our customers and partners can count on.

- With Snowflake at our core, we believe that good data has the power to enable innovation without limits, helping you say goodbye to complex data solutions and hello to the modern world of cloud elasticity.
- With an average tenure of 23 years in the big data, integration, and technology industries, our team understands the importance of data to drive informed decisions and enhanced perspective on your most critical business processes.

3. SCHEDULE OF THE TRAINING (INTERNSHIP)

Duration: 03 Months (From 24/01/2022 to 25/04/2022)

The online training / internship was carried out as per the below day wise schedule starting from 24/01/2022 to 25/04/2022.

4. TRAINING/INTERNSHIP ACTIVITIES:

4.1 Prepare Your Lab Environment

The Snowflake edition (Standard, Enterprise, e.g.), cloud provider (AWS, Azure, e.g.), and Region (US East, EU, e.g.) should be selected. We should select the region which is physically closest to us. And select the Enterprise edition so we can leverage some advanced capabilities that are not available in lower Editions.

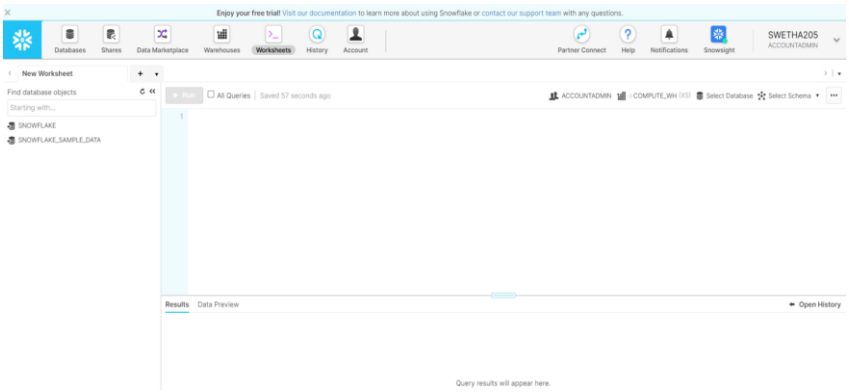
After registering, you will receive an email with an activation link and your Snowflakeaccount URL. Bookmark this URL for easy, future access. After activation, you will create a user name and password. Write down these credentials so that we can refer it if forgotten.

4.2 The Snowflake user Interface and lab "story"

4.2.1. Logging Into the Snowflake User Interface (UI): Open a browser window and enter the URL of the Snowflake 30-day trial environment.

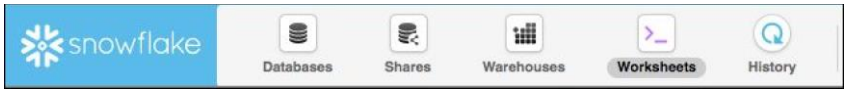
You should see the login screen below. Enter your unique credentials to log in.

4.2.2 Overview of Web User Interface: You will be able to see the following web interface when logged in using credentials.

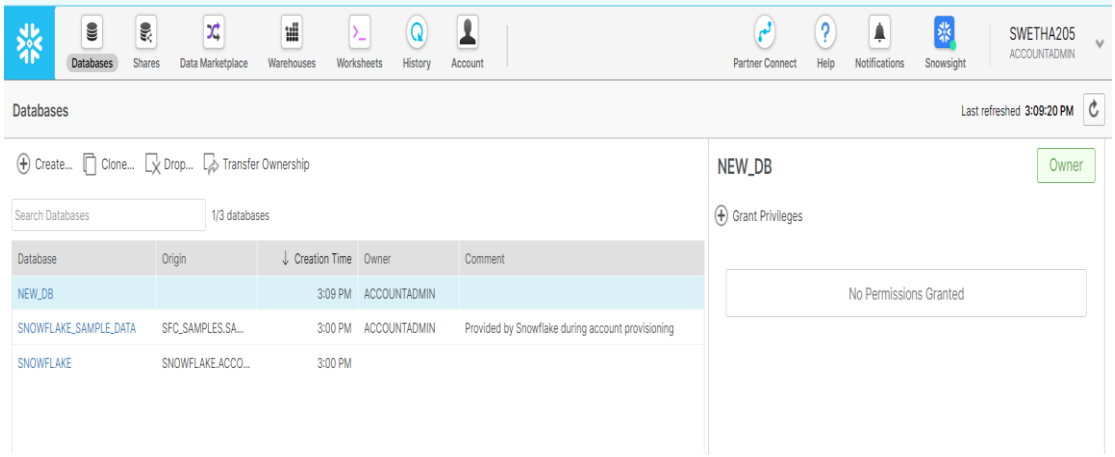


4.2.3 Navigating the Snowflake UI: First let's get to know about Snowflake! This section covers the basic components of the user interface help us to understand.

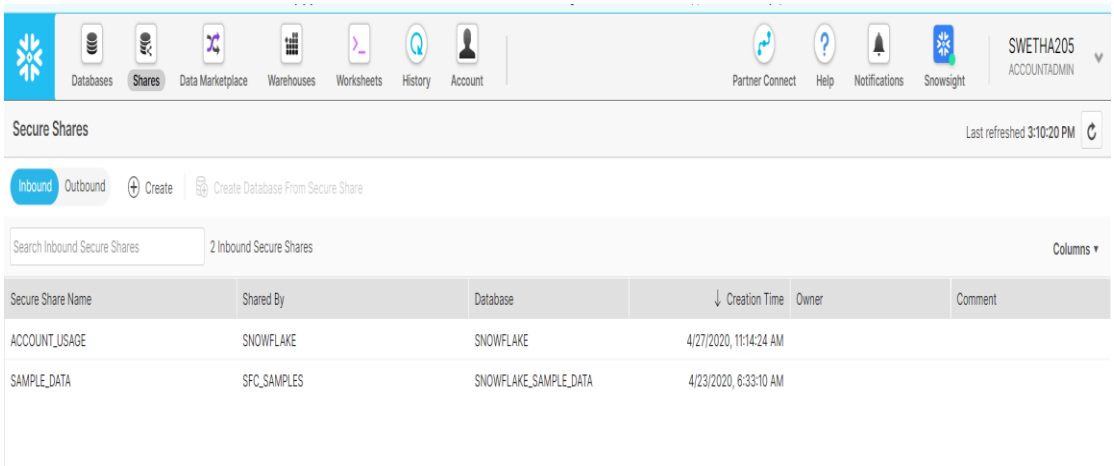
The top menu allows you to switch between the different areas of Snowflake:



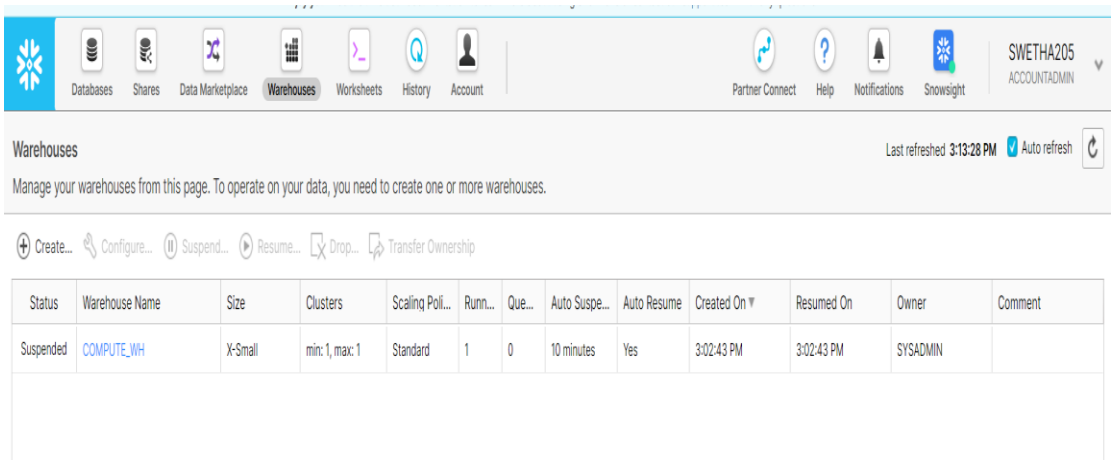
The Databases tab shows information about the databases you have created or have privileges to access. You can create, clone, drop, or transfer ownership of databases aswell as load data (limited) in the UI. Notice that one database is created.



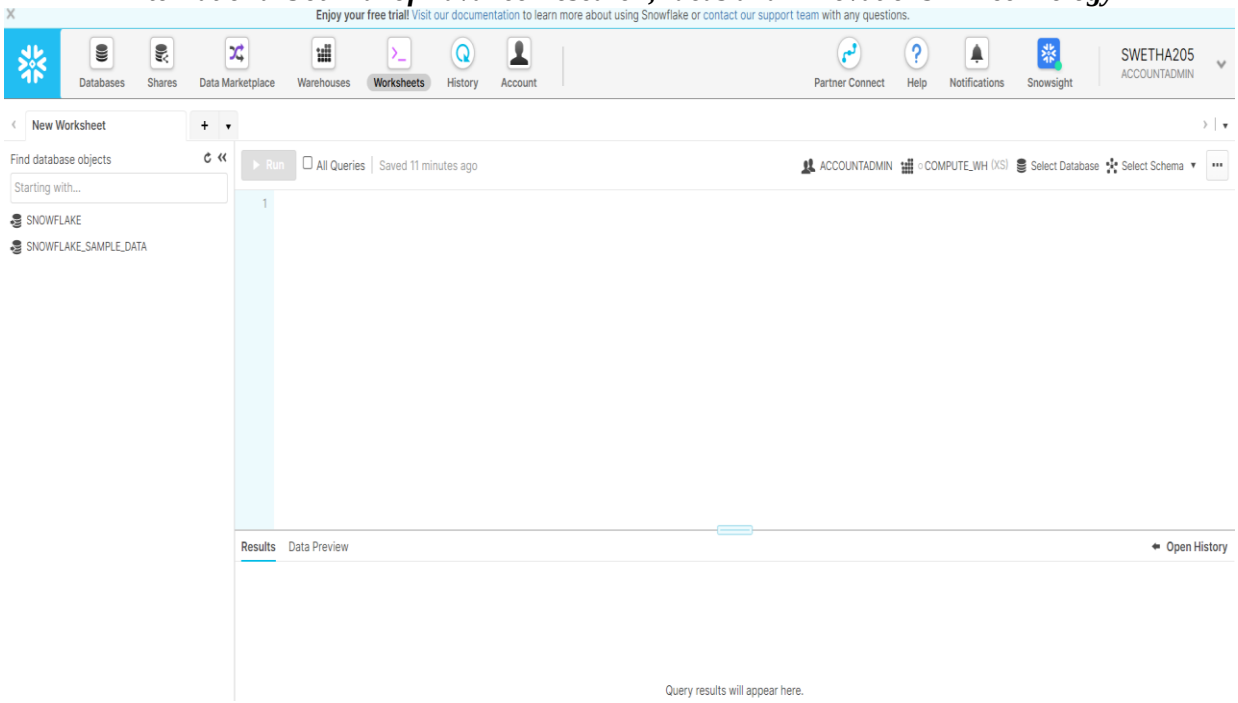
The Shares tab is where data sharing can be configured to easily and securely shareSnowflake table(s) among separate Snowflake accounts or external users, without having to create a second copy of the table data. We can create new share by clicking create option shown in the figure.



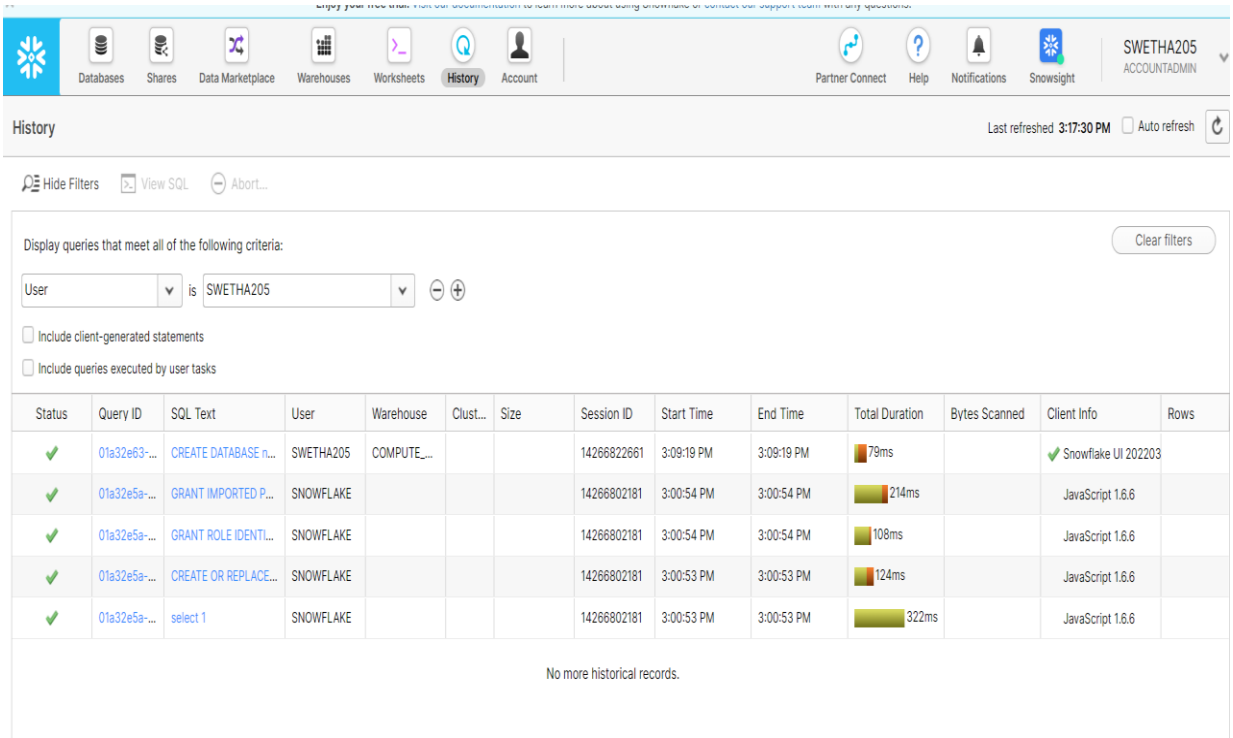
The Warehouses tab is where we create, set up and manage compute resources (virtualwarehouses) to load or query data in Snowflake. Note a warehouse called “COMPUTE_WH (XL)” already exists in your environment.



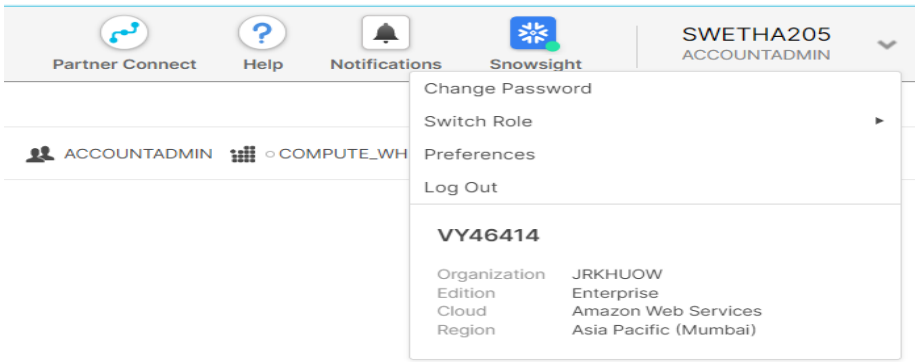
The Worksheets tab provides an interface for submitting SQL queries, performing DDLand DML operations and viewing results as your queries/operations complete. The default “New Worksheet” appears. In the left pane is the database objects browser which enables users to explore all databases, schemas, tables, and views accessible by the role selected for a worksheet.The bottom pane shows results of queries and operations.



The History tab allows you to view the details of all queries that are executed in the last 14 days in the Snowflake account.



If you click on the top right of the User Interface(UI) where your user name appears, you will see that here you can do things like change your password, roles, or preferences. Snowflake has several system defined roles like AccountAdmin, SysAdmin, UserAdmin , SecurityAdmin, Public . You are currently in the default role of ACCOUNTADMIN.



4.3 Roles and privileges

We have some key concepts like:

- Securable Object: An entity to which access can be granted.
- Role: An entity to which privileges can be granted.
- Privilege: A defined level of access to an object.
- User: A user identity recognized by Snowflake, whether associated with a person or program.

We have system defined roles provided by snowflake (Account Admin, Sys Admin, User Admin, Security Admin, Public) as well as snowflake allows us to create our own roles called as custom roles.

There may be many users under a role. We have securable objects like database, schema and tables. In order to access these objects, role must have privilege on the particular object.In this we will create different roles, users, objects like database, warehouse and assigned privileges to the created roles and users on particular objects for them to access.

```
create or replace role relationshipmanager;
create or replace role budgetanalyst;
create or replace role assetmanager;

grant role relationshipmanager to role sysadmin;
grant role budgetanalyst to role sysadmin;
grant role assetmanager to role sysadmin;


show roles;

create warehouse wh;

create or replace warehouse wh1;
grant all on warehouse wh1 to role relationshipmanager;

create or replace warehouse wh2;
grant all on warehouse wh2 to role budgetanalyst;

create or replace warehouse wh3;
grant all on warehouse wh3 to role assetmanager;

create or replace database bankorg;

create or replace schema bankorg.relationshipmanager;
create or replace schema bankorg.budgetanalyst;
create or replace schema bankorg.assetmanager;

use role useradmin;


create or replace user re_user password='rt&_123' default_role='sysadmin', default_namespace='bankorg.rm' default_warehouse='wh1' must_change_password=false;
create or replace user bu_user password='byu89' default_role='sysadmin' default_namespace='bankorg.bm' default_warehouse='wh2' must_change_password=false;
create or replace user as_user password='kilo678' default_role='sysadmin', default_namespace='bankorg.am' default_warehouse='wh3' must_change_password=false;


use role securityadmin;
grant role relationshipmanager to user re_user;
grant role  budgetanalyst to user bu_user;
grant role  assetmanager to user as_user;

show roles;
```

4.4 Loading data into table using web user interface

We can insert data into table using “INSERT” query statement but in real world there will be huge amounts of data to be inserted into table, Snowflake makes this process easier if the following steps are followed.

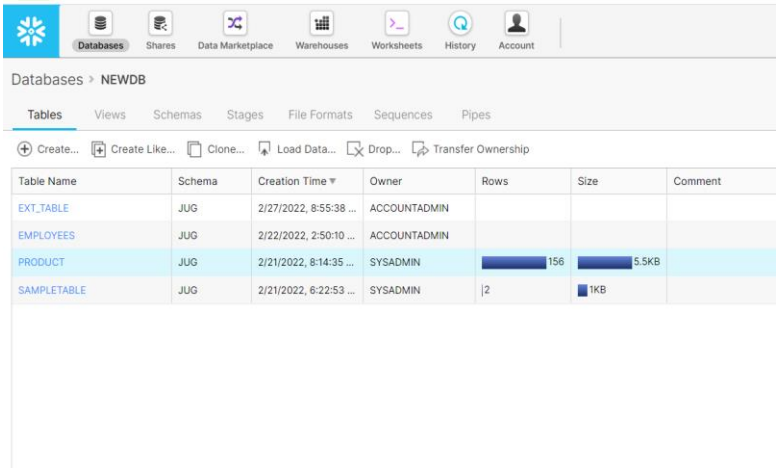
(a) Create a Table, database and schema:

```
create database newdb;
use newdb;
create schema jug;
use schema jug;


use database newdb;


create or replace table Product (
   Id                  number              identity,
   ProductName         varchar(50)         not null,
   SupplierId          number              not null,
   UnitPrice           float               null default 0,
   Package             varchar(30)         null,
   IsDiscontinued      boolean             not null
);
```

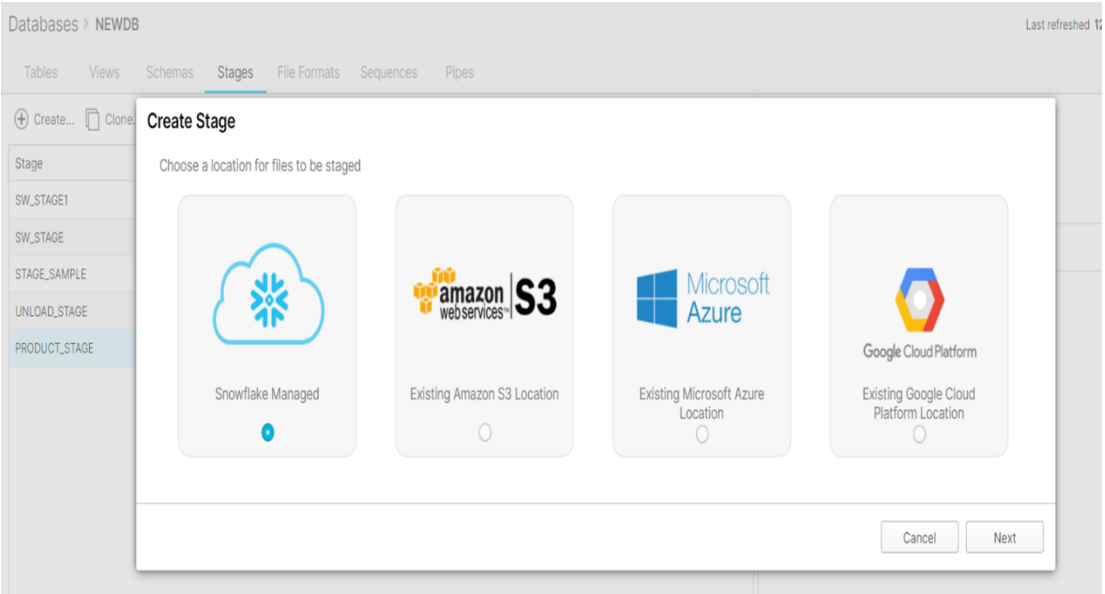
We will be using the database and schema that we have created.

(b) Create a stage

Click on Databases -> select database created -> under that select table name ->now click on stage and create one.

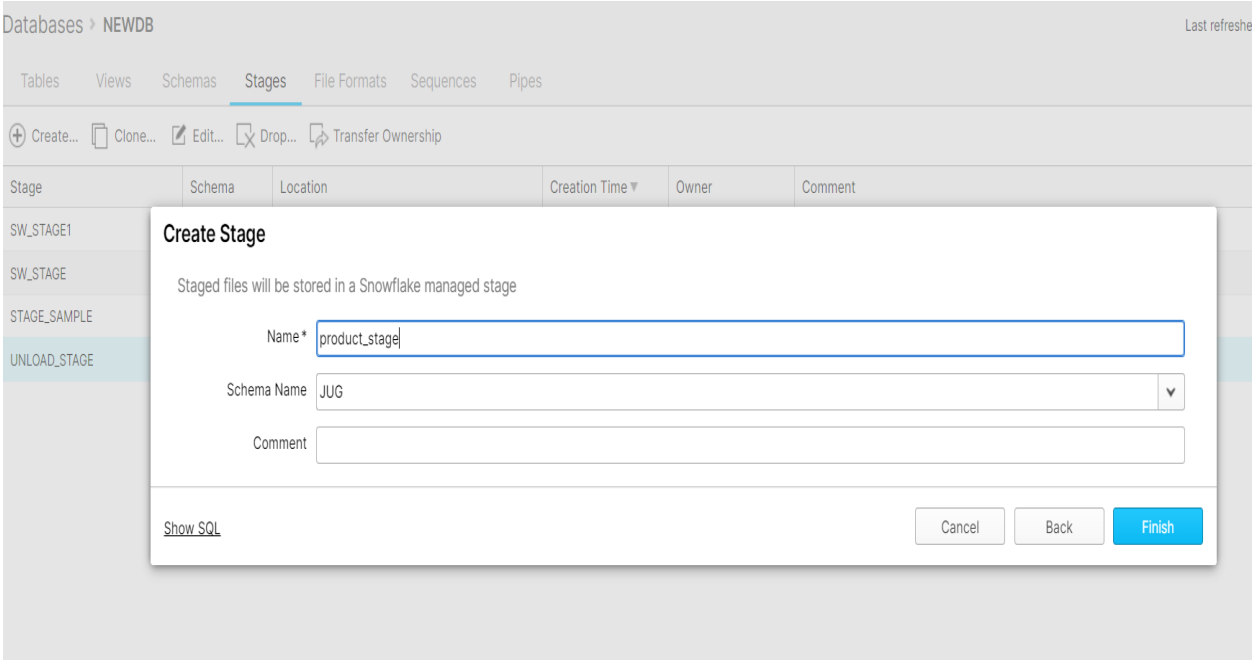


We need to click on stages and the stages can be created.



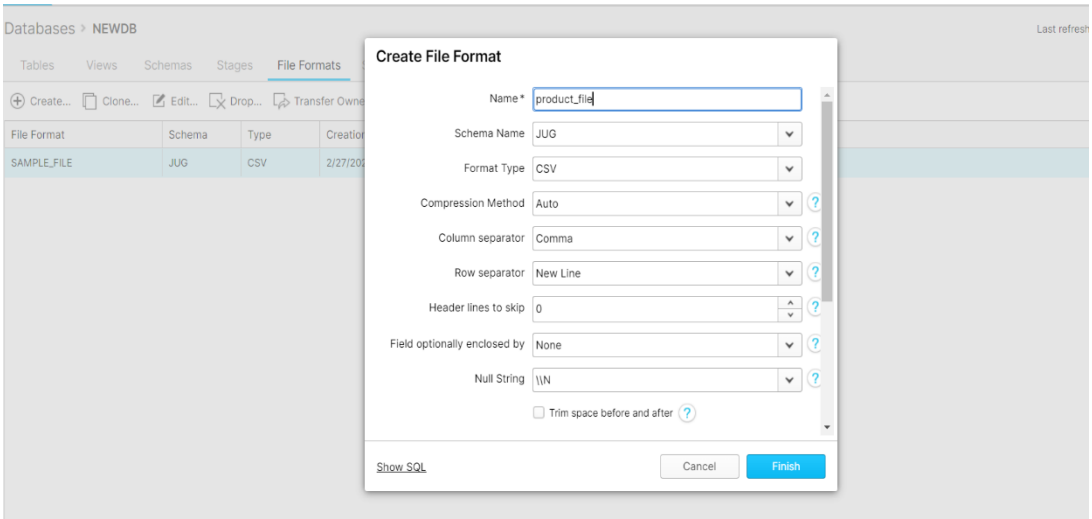
Select snowflake managed and clicks next.

Enter the stage name and select schema name and click Finish button.



(c) Create a File Format:

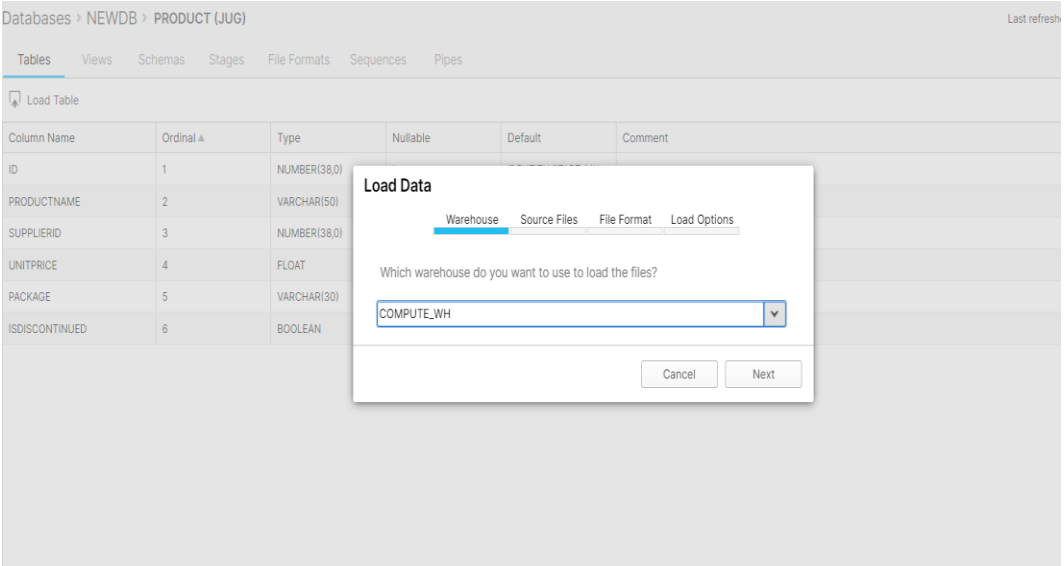
Click on Databases -> select database created -> under that select table name ->now click on file format and create one.



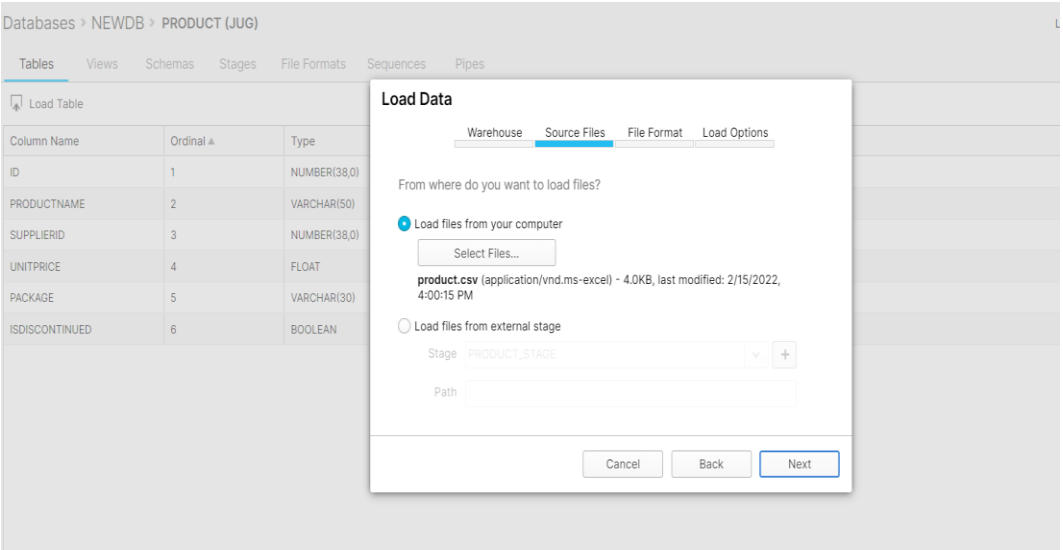
We will create file format and select schema and format type and click create button.

(d) Loading the data:

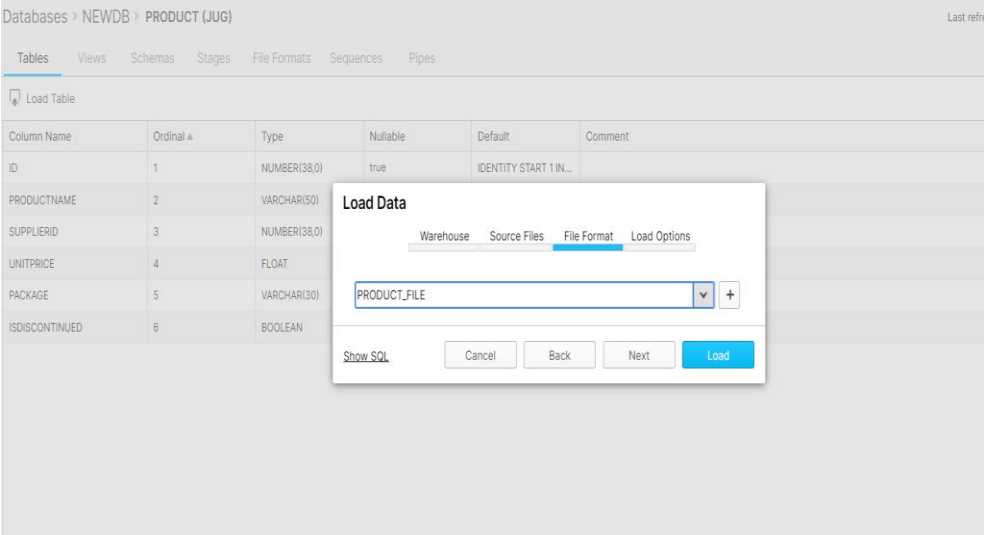
Click on databases->click on created database -> click on the table created -> select load Table option shown in figure below. In the figure below we need to select appropriate warehouse and click next button



Now, we need to upload csv. file from local system as shown in the figure below and click next button

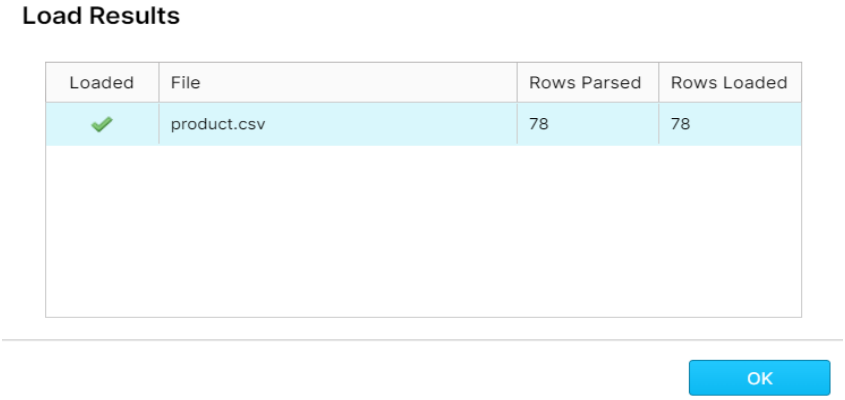


Now, select file format and click next button as shown in the figure below

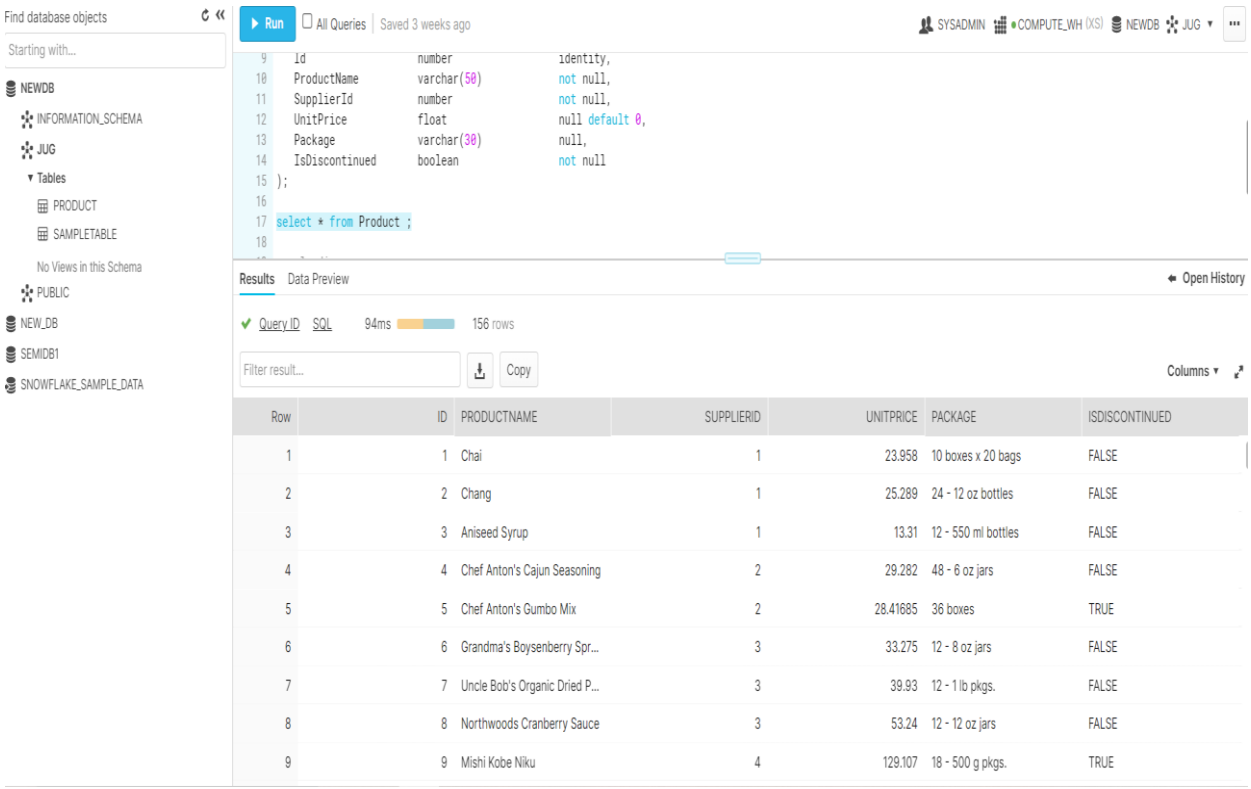


Now in the next pop-up page leave the option as default and click on load option.

In the figure below, it is shown as data loaded



Once the data has been loaded into table.
We can now be able to display data from table using select statement.
Command: select * from product;



4.5. Dynamic Data Masking

Dynamic Data Masking is a Column-level Security feature that uses masking policies to selectively mask plain-text data in table and view columns at query time. Column-level Security in Snowflake allows the application of a masking policy to a column within a table or view.

How we will do the dynamic data masking?

- We will create database, warehouse and need to use database, warehouse. We can either create new schema or even use already available public schema.
- Create a table and let us insert some tuples into the table.
- Now we will create roles, grant privileges on database objects to roles, assigning roles to users.
- Then we will create masking policy. Masking is applied on "PHONE" column in the table.
- Then we will check whether masking policy is applied successfully or not.
- Here masking is applied on the basis of role, the particular column will be visible to the role,who is authorized to view that column or else column will not be visible.

```
//Database Creation
CREATE or replace DATABASE maskingdb;
use schema public; //using public schema
//Warehouse Creation
create warehouse mask_wh;
use warehouse mask_wh;
//Table Creation
create or replace table customers(
  id number,
  full_name varchar,
  email varchar,
  phone varchar,
  spent number,
  create_date DATE DEFAULT CURRENT_DATE);
  //Inserting data into table
insert into customers (id, full_name, email,phone,spent)
values
  (1,'Lewiss MacDwyer','lmacdwyer0@un.org','262-665-9168',140),
  (2,'Ty Pettingall','tpettingall1@mayoclinic.com','734-987-7120',254),
  (3,'Marlee Spadazzi','mspadazzi2@txnews.com','867-946-3659',120),
  (4,'Heywood Tearney','htearney3@patch.com','563-853-8192',1230),
  (5,'Odilia Seti','oseti4@globo.com','730-451-8637',143),
  (6,'Meggie Washtell','mwashtell5@rediff.com','568-896-6138',600);
select * from customers;
// Creating roles , granting privileges on datbase objects to roles, assigning roles to users
USE ROLE securityadmin;
CREATE OR REPLACE ROLE ANALYST_MASKED; // Role creation
CREATE OR REPLACE ROLE ANALYST_FULL; //Role Creation
//Granting Privileges
GRANT ALL PRIVILEGES ON DATABASE MASKINGDB TO ROLE ANALYST_MASKED;
GRANT ALL PRIVILEGES ON DATABASE MASKINGDB TO ROLE ANALYST_FULL;
GRANT SELECT ON TABLE MASKINGDB.PUBLIC.CUSTOMERS TO ROLE ANALYST_MASKED;
GRANT SELECT ON TABLE MASKINGDB.PUBLIC.CUSTOMERS TO ROLE ANALYST_FULL;
GRANT USAGE ON SCHEMA MASKINGDB.PUBLIC TO ROLE ANALYST_MASKED;
GRANT USAGE ON SCHEMA MASKINGDB.PUBLIC TO ROLE ANALYST_FULL;
GRANT USAGE ON WAREHOUSE MASK_WH TO ROLE ANALYST_MASKED;
GRANT USAGE ON WAREHOUSE MASK_WH TO ROLE ANALYST_FULL;
GRANT ROLE ANALYST_MASKED TO USER swetha255;
GRANT ROLE ANALYST_FULL TO USER swetha255;
GRANT CREATE MASKING POLICY ON SCHEMA "MASKINGDB"."PUBLIC" TO ROLE ANALYST_MASKED;
GRANT APPLY MASKING POLICY ON ACCOUNT TO ROLE ANALYST_MASKED;
USE ROLE ANALYST_MASKED;
//Creating Masking Policy
CREATE OR REPLACE masking policy phone
    AS (val varchar) RETURNS varchar ->
            CASE
            WHEN current_role() IN ('ANALYST_FULL') THEN val
            ELSE '**********'
            END;
ALTER TABLE IF EXISTS CUSTOMERS MODIFY COLUMN phone
SET MASKING POLICY PHONE; //AccountAdmin role
USE ROLE ANALYST_FULL; // Use Analyst_Full role and check
SELECT * FROM CUSTOMERS;
USE ROLE ANALYST_MASKED; // Use role Analyst_Masked role , this time phone column will be masked and will be displayed
SELECT * FROM CUSTOMERS;
```

When used ANALYST_FULL:

```
SET MASKING POLICY PHONE;//execute this using acountadmin
USE ROLE ANALYST_FULL;
SELECT * FROM CUSTOMERS_new;
USE ROLE ANALYST_MASKED;
SELECT * FROM CUSTOMERS_new;
```

Results Data Preview Open History

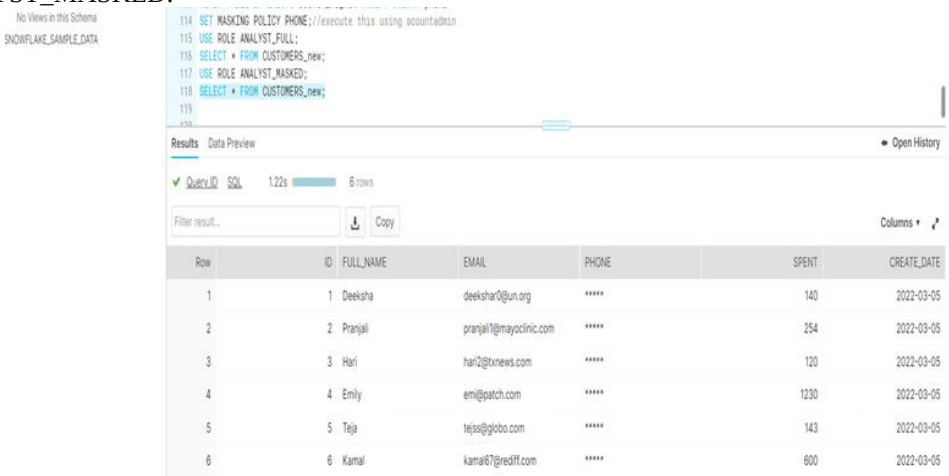
✔ Query ID SQL 100ms 6 rows

Filter result... Copy Columns

Row	ID	FULL_NAME	EMAIL	PHONE	SPENT	CREATE_DATE
1	1	Deeksha	deekshar0@un.org	262-665-9168	140	2022-03-05
2	2	Pranjali	pranjali1@mayoclinic.com	734-987-7120	254	2022-03-05
3	3	Hari	hari2@txnews.com	867-946-3659	120	2022-03-05
4	4	Emily	emi@patch.com	563-853-8192	1230	2022-03-05
5	5	Teja	tejss@globo.com	730-451-8637	143	2022-03-05

Here, phone column will be visible because we wrote in the code that phone column to be visible when used ANALYST_FULL role.

When used ANALYST_MASKED:



When this role is used phone column will be masked since as in the code when this role tries to access the table, this column should be kept secret or not visible.

4.6. Secure Data Share

Secure Data Sharing enables sharing selected objects in a database in your account with other Snowflake accounts.

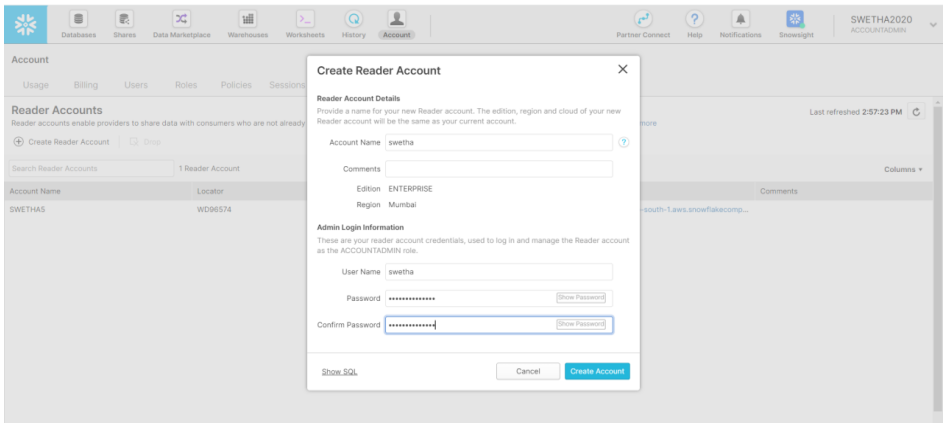
- The account that receives the data cannot modify it, as shared data is always read-only.
- These are the Snowflake objects that we can share: Tables, External tables, Secure views, Secure materialized views.
- Data sharing takes place between Producer and Consumer.

There are two types of consumers:

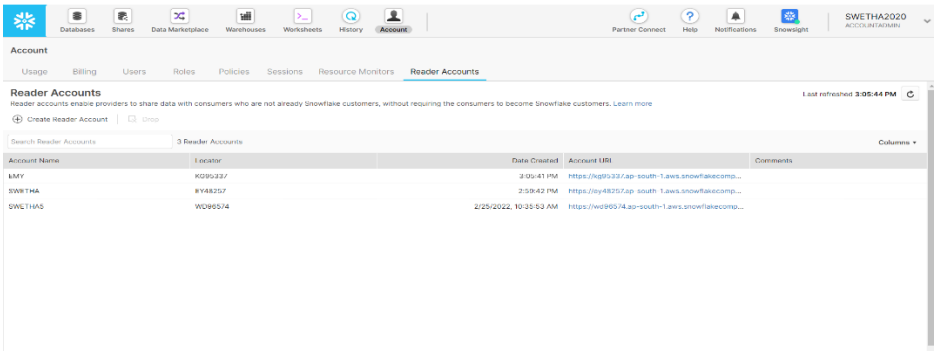
1. Full Account: When you share something with an existing SnowFlake account.
2. Reader Account: This account is used to share data with someone who has no Snowflake account.

PRODUCER

Step1: Producer will create reader account. We will create reader account (consumer) in order to share data. We have Account option under AccountAdmin role. Under Account option click on Reader account and click create reader account. [Here we will share data to Reader account].

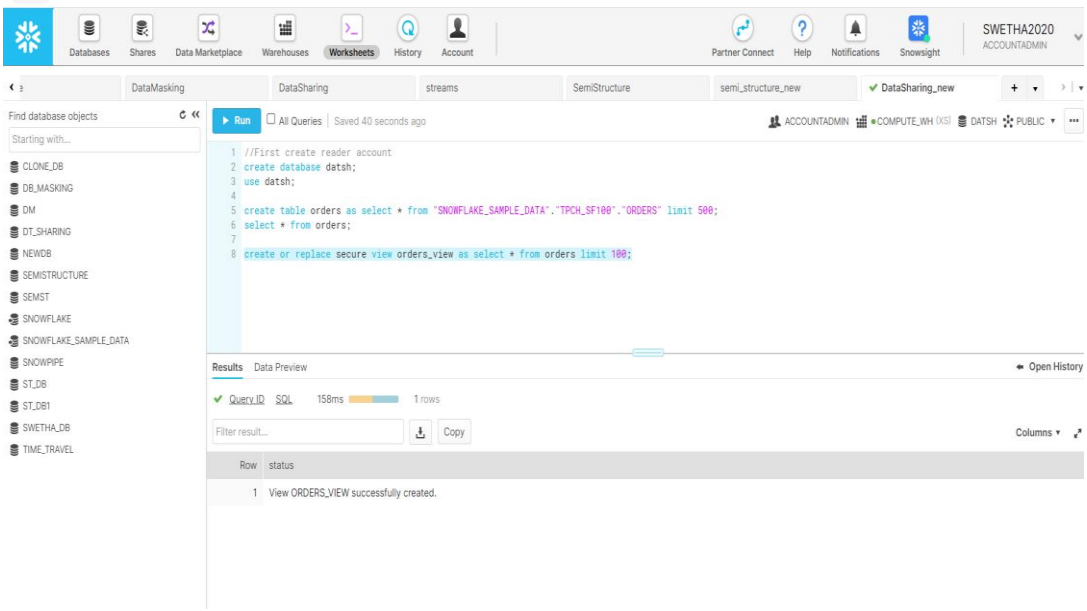


Here we need to give account name, username and password and click create account button.

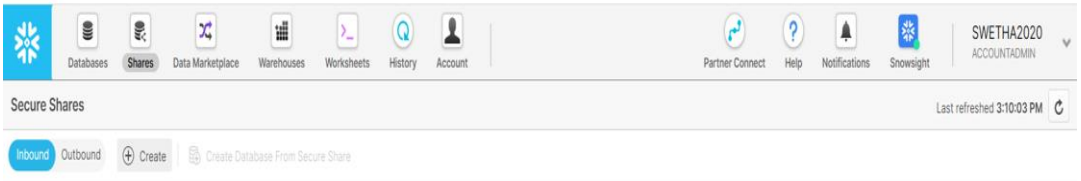


We can login into any of the reader account (consumer) using the Account URL. We should enter the credentials and log in
We will be sharing secure views, so we need to create secure view.

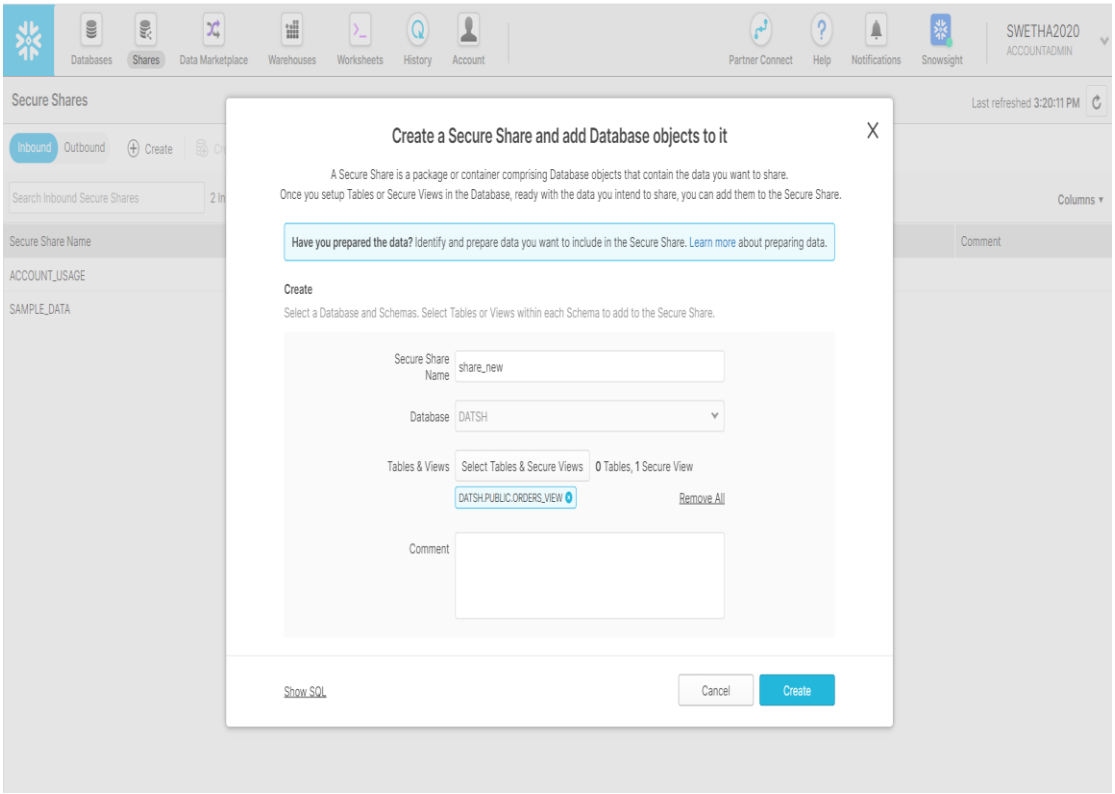
Create or replace secure view orders view as select * from orders limit 100;



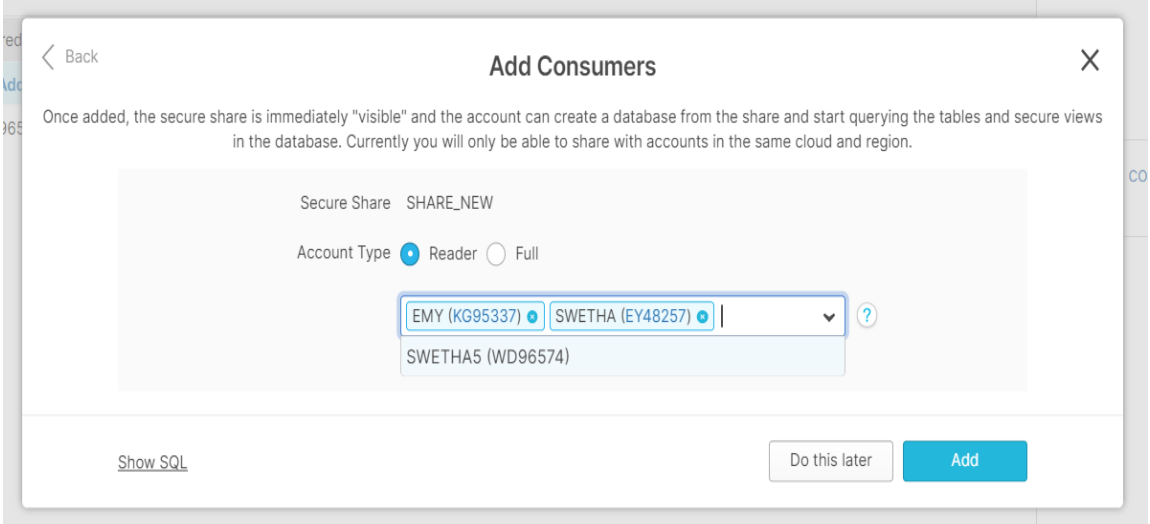
Now let us select the objects to be shared with consumer:



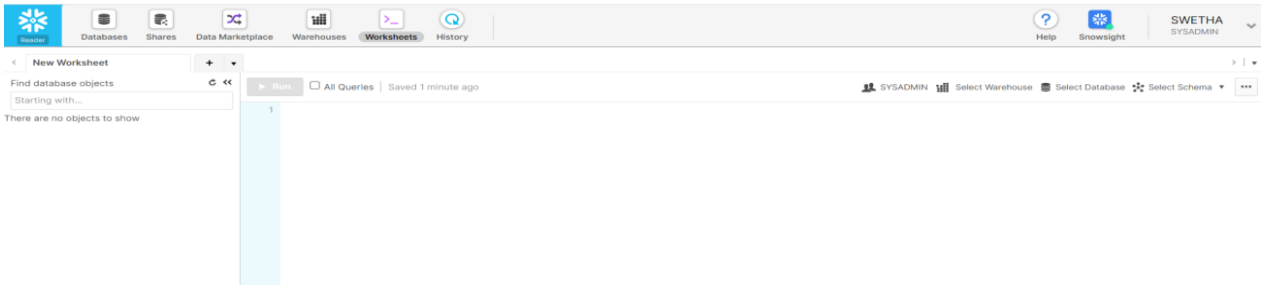
Click on share option from the above figure.



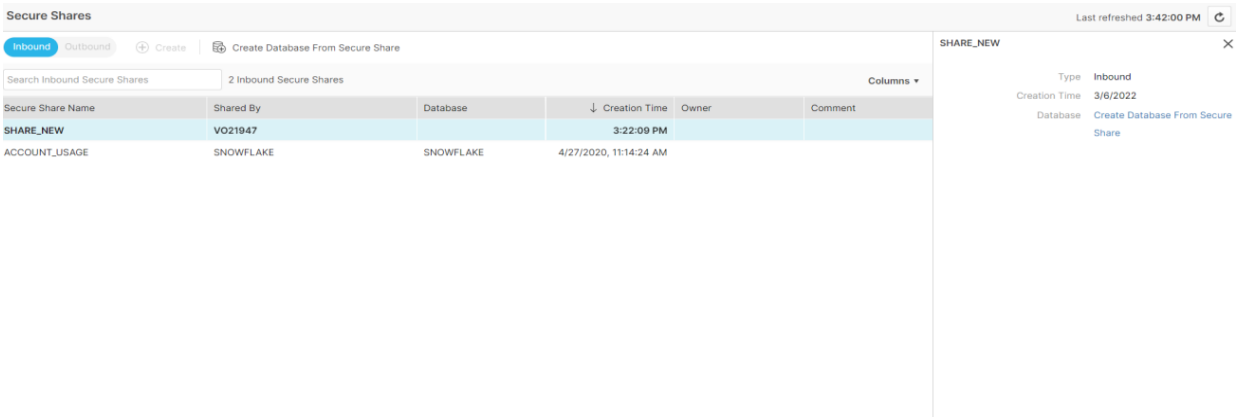
We need to give share name and select view that we want to share with consumer.
After clicking create button we need to add consumer names with whom data needs to shared and click Add button.

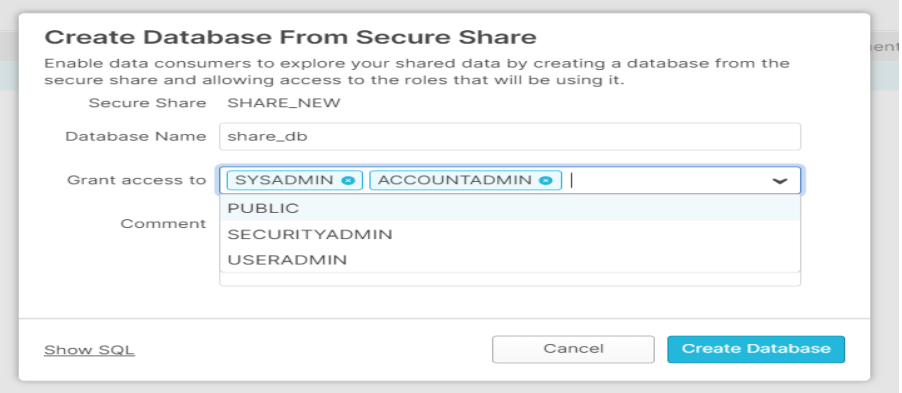


Now, login as any of the reader account using credentials. over there we need to create one database and through that we can access the shared view.

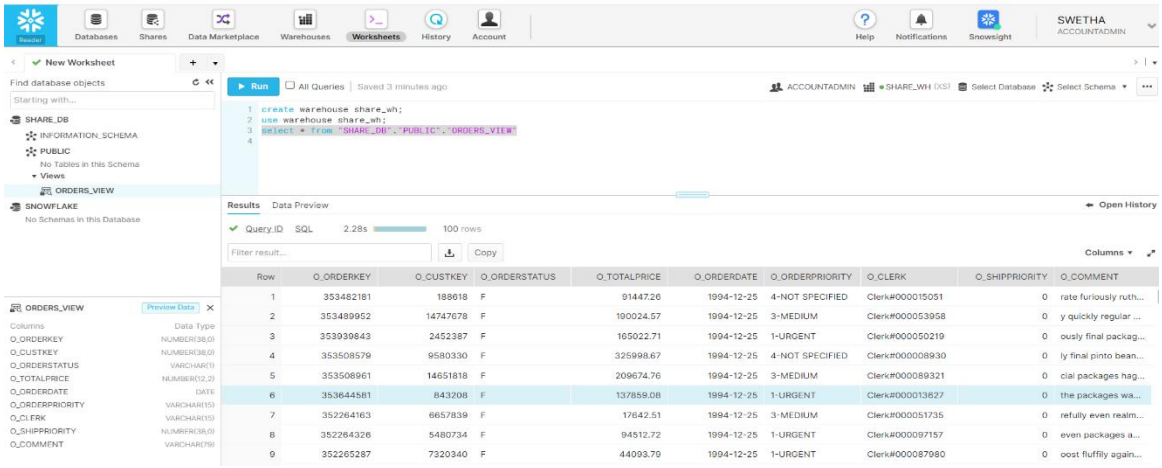


Change account to account admin role and click on account we will have share option and click on it , then click on created share , we will get an option to create database.





Click on create database.
Now, we need to create database and warehouse and use select query to display data from view.

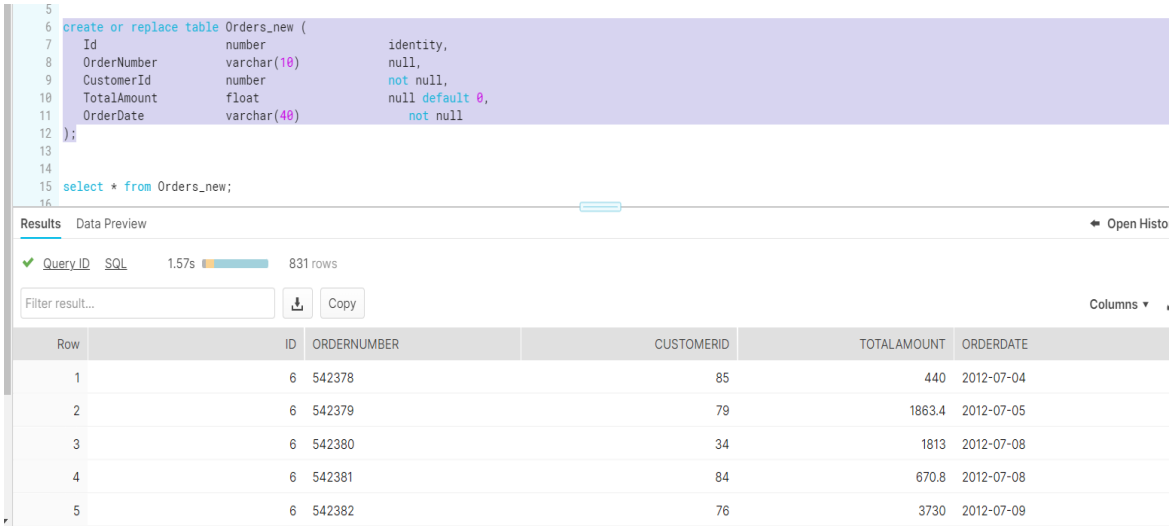


4.7. Time Travel

- Time travel enables accessing historical data (i.e., data that has been changed or deleted) at any point within a defined period.
- Use cases of Time Travel:
- Restoring data-related objects that we might have accidentally or intentionally deleted.
- Query data from the past.

Now we will create one table, also having some data and try to query data and restoring data from past.

(a) Table Creation



Restoring the object: To restore objects, we use the command "UNDROP". We can use it with Databases, Schemas, or Tables.

Use undrop command to restore the table object

```
32
33 undrop table Orders_new;
34
35 --Data Backup
36 create or replace table order_backup as select * from Orders_new before (statement=>'01a27ef3-0000-2075-0000-0002f3f22c45');
```

Results Data Preview

Query ID SQL 47ms 1 rows

Filter result... Copy

Row	status
1	Table ORDERS_NEW successfully restored.

• Querying over historical data:

1. By Offset: First we will update the table, the query the table.
update orders_new set ID=9; // Updating the table

Command:
->select * from Orders_new at (offset => -60*2);
The above statement will show the old table data, i.e., the table before updation.

```
17 update orders_new set ID=9;
18
19 select * from Orders_new at (offset => -60*2);
20
21
```

Results Data Preview Open History

Query ID SQL 85ms 830 rows

Filter result... Copy Columns

Row	ID	ORDERNUMBER	CUSTOMERID	TOTALAMOUNT	ORDERDATE
1	1	542378	85	440	2012-07-04
2	2	542379	79	1863.4	2012-07-05
3	3	542380	34	1813	2012-07-08
4	4	542381	84	670.8	2012-07-08
5	5	542382	76	3730	2012-07-09
6	6	542383	34	1444.8	2012-07-10

2. By query statement ID: The table will contain updated values. In order to access the old table we can also retrieve it using statement ID.

```
19 select * from Orders_new at (offset => -60*5);
20
21 select * from Orders_new;
22
23
```

Results Data Preview Open History

Query ID SQL 29ms 830 rows

Filter result... Copy Columns

Row	ID	ORDERNUMBER	CUSTOMERID	TOTALAMOUNT	ORDERDATE
1	9	542378	85	440	2012-07-04
2	9	542379	79	1863.4	2012-07-05
3	9	542380	34	1813	2012-07-08
4	9	542381	84	670.8	2012-07-08
5	9	542382	76	3730	2012-07-09
6	9	542383	34	1444.8	2012-07-10
7	9	542384	14	625.2	2012-07-11

The query Id belongs to the query ID of Update statement, we will get original table instead of new table.


```
update orders_new set ID=9;

select * from Orders_new at (offset => -60*5);

select * from Orders_new before(statement=>'01a328ad-0000-242b-0000-00032122b141');
```

Results Data Preview Open History

Query ID SQL 91ms 830 rows

Filter result... Copy Columns

Row	ID	ORDERNUMBER	CUSTOMERID	TOTALAMOUNT	ORDERDATE
1	1	542378	85	440	2012-07-04
2	2	542379	79	1863.4	2012-07-05
3	3	542380	34	1813	2012-07-08
4	4	542381	84	670.8	2012-07-08
5	5	542382	76	3730	2012-07-09
6	6	542383	34	1444.8	2012-07-10
7	7	542384	14	625.2	2012-07-11

5. OUTCOMES

The main objective of this technology is to help students, trainees understand the basics of Snowflake cloud data warehouse. Looking at Snowflake's history as a data warehouse and the features that come along with it, it is easy to see what makes Snowflake best in class. A winning formula comprised of unmatched performance, cloud-focused architecture, ease of use, and intuitive pricing have all set the stage for where Snowflake can go next. Snowflake has won the cloud data warehouse battle—for years now, competitors have been chasing features that Snowflake launched with, and they can't catch up. These themes that set Snowflake apart as a data warehouse will, without a doubt, be the pillars upon which they expand the data platform to new areas. The power of Snowflake's architecture isn't unique to the data warehouse; it really is the catalyst to creating a data platform that organizations can rely on and move into the future with.

Moreover, with snowflake

- Data loading has been simpler as we have seen.
- Even our data can be masked which will help us to protect sensitive data.
- Secure Data Sharing is made easier.
- We can able to restore the dropped table, database or schema if it is deleted due to human error or unintentionally.

7. REFERENCES:

Journals/Articles/Bibliography:

[1] https://docs.snowflake.com/en/
[2] https://medium.com/default-to-open/snowflake-the-details-of-our-first-data-warehousing-project-in-the-cloud-2e5d4435275c
[3] https://www.projectpro.io/project-use-case/snowflake-real-time-data-warehouse-project-for-beginners

Thesis:

[1] http://www.met.reading.ac.uk/~sws04cdw/thesis.pdf

Books:

[1] https://www.oreilly.com/library/view/snowflake-the-definitive/9781098103811/
[2] https://www.snowflake.com/resource/cloud-data-warehousing-dummies/

